第1章 数制和码制

东南大学电气工程学院

第1章 数制和码制

- 1.1 计算机中的数制
- 1.2 计算机中数的表示方法
- 1.3 非数值数据在计算机中的表示方法

1.1 计算机中的数制

在模拟电子电路中，要处理的是连续的模拟信号，采用如微分方程、拉氏变换这类表达连续量及其关系的数学工具。

在数字电子电路中，要处理的是离散的二值逻辑量，采用的数学工具是逻辑代数（布尔代数）

如何用逻辑量来表示数字？？

——数制与码制

数制与码制分别是计数制与编码制的简称.

1.1 计算机中的数制

1. 几种常用的计数制
   - 十进制
   - 二进值
   - 八进制
   - 十六进制

R进制的幂级数展开式

$$(N)_R = a_{n-1}a_{n-2}\cdots a_2a_1a_0$$

$$= a_{n-1}\times R^{n-1} + a_{n-2}\times R^{n-2} + \cdots + a_2\times R^2 + a_1\times R^1 + a_0\times R^0$$

$$= \sum_{i}^{n-1} a_i R^i$$

1.1 计算机中的数制

2. 不同数制之间的相互转换

(1)二、八、十六进制转换成十进制

例1 将（11010）$_2$ 转换为十进制。

$$（11010）_2 = 1\times2^4 + 1\times2^3 + 1\times2^1 = （26）_{10}$$

例2 将（274）$_8$ 转换为十进制。

$$（274）_8 = 2\times8^2 + 7\times8^1 + 4\times8^0 = （188）_{10}$$

1.1 计算机中的数制

(2)十进制数转换成二、八、十六进制数

纯整数：除基取余法

余数

```
2 | 25   1
2 | 12   0
2 |  6   0
2 |  3   1
2 |  1   1
    0
```

$(25)_{10}=(11001)_2$

纯小数：乘基取整法

整数

```
0.125
 ×   2
0.250   0
 ×   2
0.500   0
 ×   2
1.000   1
```

$(0.125)_{10}=(0.001)_2$

1.1 计算机中的数制

(3)二进制数与八、十六进制数的互换

二-八互换 ： 三位二进制数对应一位八进制数

例：$(367.505)_8=(011110111.101000101)_2$

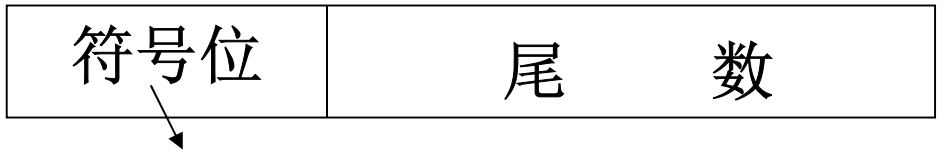
$(11101.01011)_2=(35.26)_8$

二-十六互换 ： 四位二进制数对应一位十六进制数

1.2 计算机中数的表示方法

在计算机中对数据进行运算操作时,将符号位和数值位在一起编码来表示相应的数----机器码. 而一般书写表示的数----真值.

机器码 { 原码 / 补码 / 反码



0 表示 “+”、1 表示 “-”

实数在计算机中的表示

（1）原码

真值	原码
$x = +x_1x_2\cdots x_{n-1}$	$[x]_{原} = 0x_1x_2\cdots x_{n-1}$
$x = -x_1x_2\cdots x_{n-1}$	$[x]_{原} = 1x_1x_2\cdots x_{n-1}$

实数在计算机中的表示

$$[x]_{\text{原码}} = \begin{cases} x & 0 \le x \le 2^{n-1} - 1 \\ 2^{n-1} + |x| = 2^{n-1} - x & -2^{n-1} + 1 \le x \le 0 \end{cases}$$

例，对于**8**位二进制原码，其有效位数为**7**位表示范围：

-127～+127

实数在计算机中的表示

（2）反码

反码定义如下：

*对于正数，它的反码与原码相同。

*对于负数，除符号位仍为“1”，其余各位取反。

实数在计算机中的表示

（3）补码

补码技术实际上是用机器来处理负数的技术。

A-B=A-B+M=A+（M-B）=A+B’

----B’为（-B）的补数

*式中**B>0**.对于代数式**A+B, M+B**还是**B**,不影响**A+B**的运算.所以正数的补数仍是它本身.

解决求补困难的问题。比较原码与补码的尾数部分可以发现，原码每一位数码取反后再加**1**即为它的补码。

实数在计算机中的表示

设$x = -x_1 x_2 \cdots x_{n-1}$

* 在机器中,n位二进制补码的符号位S的规定与原码相同.

负数的补码，尾数$m' = 2^{n-1} - |x|$

*考虑符号位一起参加运算：　则：$[x]_{补} = 2^n - |x|$

*n位补码可表示的真值范围是$[-2^{n-1}, 2^{n-1} - 1]$

实数在计算机中的表示

设 $x = -x_1 x_2 \cdots x_{n-1}$ 则 $[x]_{反} = 1\overline{x_1}\,\overline{x_2} \cdots \overline{x_{n-1}}$

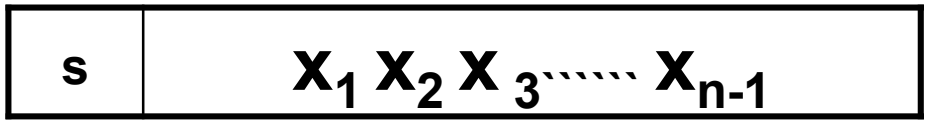
$$|x| + [x]_{反} = \underbrace{111\cdots1}_{n位} = 2^n - 1$$

则 $[x]_{反} = (2^n - 1) - |x|$ （1’s补码）

显然：$[x]_{补} = [x]_{反} + 1$

定点数在计算机中的表示

定点数：小数点位置固定



对于纯小数，默认小数点位于s 与x_1之间。

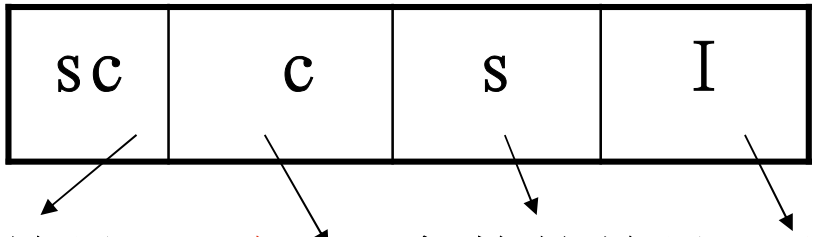
可表示的范围为 $-(1-2^{-(n-1)}) \sim +(1-2^{-(n-1)})$

对于纯整数，默认小数点位于x_{n-1}的右边。

可表示的范围为 $-(2^{n-1}-1) \sim +(2^{n-1}-1)$

浮点数在计算机中的表示

浮点数：小数点位置不定



阶码的符号　阶码　实数的符号　尾数（整数）

k位　m位

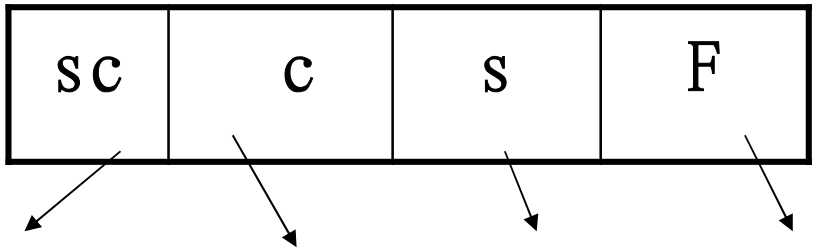
表示的数为

$$V=(-1)^{s}\times I\times 2^{(-1)^{sc}\times c}$$

表示的范围为

$$-\left(2^{m}-1\right)\times 2^{2^{k}-1}\sim+\left(2^{m}-1\right)\times 2^{2^{k}-1}$$

浮点数在计算机中的表示



表示的数为

$$V=(-1)^{s}\times 0.F\times 2^{(-1)^{sc}\times c}$$

表示的范围为

$$-(1-2^{-m})\times 2^{2^{k}-1}\sim +(1-2^{-m})\times 2^{2^{k}-1}$$

1.3 非数值型数据在计算机的表示

1. 编码制

编码制是数字电路中用符号**0**、**1**的组合来表示数字、字符等不同的对象。不同的组合就形成不同的的编码。如二进制码、循环码。

注意：码制没有数量大小的含义。

常见的编码方式

十进制符号	二进制码	四位循环码	十进制符号	二进制码	四位循环码
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

常用的四位二-十进制码（BCD码）

十进制符号	8421BCD码	余3BCD码	格雷BCD码
0	0000	0011	0000
1	0001	0100	0001
2	0010	0101	0011
3	0011	0110	0010
4	0100	0111	0110
5	0101	1000	0111
6	0110	1001	0101
7	0111	1010	0100
8	1000	1011	1100
9	1001	1100	1101

关于码制

*一个码字是由若干信息位（**bit**）组成，每位使用**0**和**1**两种代码（又叫码元），**n**位代码可表示2^n种不同信息或数据。

*一个代码的字长可以是**8**位、**16**位、**32**位、**64**位**...**，也可以以字节（**byte**）为单位，每**8**位为一个字节。

* 一个代码有时有数的概念，有时则完全没有数的概念，在数字设备中，用它可以表示任何信息。

作业：

- 1.4（3）
- 1.5（4）
- 1.9（2）
- 1.10　　(2)　(4)
- 1.12　　(3)
- 1.15　　(1)（6）（8）